Utility Theory-Based User Models for Intelligent Interface Agents

Proceedings of the Twelfth Canadian Conference on Artificial Intelligence (AI '98)
Vancouver, British Columbia, Canada, 18-20 June 1998

Scott M. Brown, Eugene Santos Jr., and Sheila B. Banks

Department of Electrical & Computer Engineering
Air Force Institute of Technology
2950 P Street, Wright-Patterson AFB, OH 45433-7765

Utility Theory-Based User Models for Intelligent Interface Agents

Scott M. Brown[1], Eugene Santos Jr.[2], and Sheila B. Banks[1]

[1] Department of Electrical and Computer Engineering
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433–7765 USA
{sbrown,sbanks}@afit.af.mil
[2] Computer Science and Engineering
University of Connecticut
Storrs, CT 06269–3155 USA
eugene@eng2.uconn.edu

Abstract. An underlying problem of current interface agent research is the failure to adequately address effective and efficient knowledge representations and associated methodologies suitable for modeling the users' interactions with the system. These *user models* lack the representational complexity to manage the uncertainty and dynamics involved in predicting user intent and modeling user behavior. A utility theory-based approach is presented for effective user intent prediction by incorporating the ability to explicitly model users' goals, the uncertainty in the users' intent in pursuing these goals, and the dynamics of users' behavior. We present an interface agent architecture, CIaA, that incorporates our approach and discuss the integration of CIaA with three disparate domains — a probabilistic expert system shell, a natural language input database query system, and a virtual space plane —that are being used as test beds for our interface agent research.

Keywords: cognitive modeling, uncertainty, knowledge representation

1 Introduction

As computers have become common place in the business work force and at home, researchers and the software industry have become painfully aware of the need to help users perform their every day tasks. To that end, research continues into *interface* or "personal assistant" agents. The purpose of these agents is to reduce information overload by collaborating with the user, performing tasks on the users behalf [28]. Examples of interface agents include office assistance agents, such as e-mail, scheduling, and financial portfolio management agents [28, 38], tutor and coach agents [11, 12], and character-based assistants for word processors, spreadsheets, and presentation software, such as the Office Assistants found in the Microsoft's Office 97 software [20].

The remainder of this paper is organized as follows: The motivation and background information for our current research efforts is provided in Section 2. Section 2.1 briefly outlines requirements we believe interface agents must meet, metrics to measure those requirements, and a methodology for determining if an agent is meeting those requirements. In Section 3, we discuss our utility theory-based approach to offering assistance to a user in an environment. Section 4 describes three of the environments we are using as test beds for our research. In Section 5, we discuss research related to our work. Finally, we discuss some pertainent issues, and drawing conclusions from our research and showing promising areas for future research.

2 Background

Shifting responsibility for final design decisions from the initial software developers to the users or to people who are closer to the users has long been realized as a desirable goal for software systems. This approach has been central to the research performed by the human-computer interaction (HCI) community. *Customizable* systems allow end users to adjust a system to their specific needs and tasks. These adjustments can be as simple as allowing a user to choose from predetermined alternatives to providing users a way to alter the system itself. Adaptive interfaces are one customization technique, but one where the user is not in complete control. Adaptations are typically done either via statistical averages of users' actions or dynamic user models (see Baeker, et. al [2] for examples of systems utilizing both approaches). We feel statistical approaches fail to adequately model a user's intent. In particular, situation-action pairs [27] ignore the correlation between actions (i.e., behaviors) and the goals being pursued. Situations are matched with a single action to perform given the situation. However, in many situations, a user may perform a series of actions to accompilsh some higher level goal. Simple situation-action pairs can not model situation-goal-actions.

Maes [27] discusses the basic problems Artificial Intelligence researchers have with adaptive autonomous agents, of which interface agents are a subset. The two basic problems are the following:

- **Action selection** — what to do next given time-varying goals?
- **Learning from experience** — how to improve performance over time?

To address the problem of action selection, interface agents must be capable of determining which goal the user is pursuing so the interface agent can determine what to do next. Since the purpose of interface agents is to offer beneficial assistance to the user, it is important to have a keen understanding of the goals the user is pursuing over time. We use the term *user intent* to denote the actions a user intends to perform in pursuit of his/her goal. Therefore, for an interface agent to be able to assist the user in pursuing those goals, the agent must be capable of ascribing user intent.

With respects to the action selection problem stated by Maes, we present a utility theory-based approach for user modeling. Our approach determines *what is important* to model in the domain. Our knowledge representation captures an explicit representation of users' goals within the domain, with associated metrics to determine *when* a user is pursuing those goals and *how* to offer assistance, and allows an agent to predict the user's intent. By infusing utility theory into our approach, we can not only determine the probability that assistance should be offered on the user's behalf, but also the utility of offering this assistance. Determining *when* and *how* to offer assistance is paramount to providing timely, beneficial assistance to the user.

To address the problem of learning from experience, we must determine *what* to improve over time. What should we learn from past experiences? Ascription of user intent is inherently uncertain. Due to the simple fact most environments where interface agents are utilized are very dynamic and not static, user models must be capable of adapting over time to better model the user with the domain. Due to space constraints, we can not fully address this problem here; the problem of learning from experience is addressed elsewhere [5, 10].

2.1 Interface Agent Development

We believe it is a necessity to first develop concrete, measurable requirements and then use these metrics to determine the effectiveness of an interface agent within an environment. We levy the following requirements on our agent: **adaptivity** — "the ability to modify an internal representation of the environment through sensing of the environment in order to change future sensing, acting, and reacting for the purpose of determining user intent and improving assistance", **autonomy** — "the ability to sense, act, and react over time within an environment without direct intervention", **collaboration** — "the ability to communicate with other agents, including the user, to pursue the goal of offering assistance to the user", and **robustness** — "the ability to degrade assistance gracefully." We have developed an associated set of requirement metrics to measure the effectiveness of the interface agent in meeting these requirements. For example, the *precision metric* measures the interface agent's ability to accurately suggest assistance to the user. We define our precision metric as

$$M_{precision} \triangleq \frac{number\ of\ correct\ suggestions}{number\ of\ suggestions}. \tag{1}$$

Details on the requirement metrics set may be found elsewhere [10].

3 Utility Theory-Based User Models

The elicitation, specification, design, and maintenance of an accurate cognitive user model of the user is necessary for effective ascription of user intent. The driving goal of our research is to develop a comprehensive software engineering, knowledge engineering, and knowledge acquisition methodology for Symbiotic

Information Reasoning and Decision Support (SIRDS) [5]. SIRDS requires the development of an adaptive, intelligent, learning human computer interface. Intelligent agents are a key aspect of SIRDS; they perform information fusion, analysis, and abstraction, as well as deriving information requirements and controlling information display.

User modeling is concerned with how to represent the user's knowledge and interaction within a system to adapt those systems to the needs of users. Researchers from the fields of artificial intelligence, human-computer interaction, psychology, and education have all investigated ways to construct, maintain, and exploit user models. The benefit of utilizing a dynamic user model within a system is to allow that system to adapt over time to a specific user's preferences, workflow, goals, disabilities, etc. To realize this benefit, the user model must effectively represent the user's knowledge and intent within the domain to accurately predict how to adapt the system.

Unfortunately, ascribing user intent is made difficult because many times users do not follow pre-planned goals. They perform actions that can be ascribed to one plan, and other actions that can be ascribed to another plan. As a result, observing a user's actions in an attempt to predict intention to those actions can be next to impossible. One way of avoiding this is to observe only the most recent actions [16]. A *fading function* is used to "forget" past actions. Not only does the fading function have the advantage focusing attention on the most recent actions making prediction of user intent in certain domains (e.g., web browsing) easier, but it has the side effect of reducing the complexity of reasoning over all the past actions to determine a user's intent. Another way to handle this problem is to introduce uncertainty into the model. Jameson describes how a causal planning model can be used to construct Bayesian networks [24]. The networks are constructed based on the observable events within each phase of the model.

One failing of purely statistical approaches is their inability to determine the utility of offering assistance for an action. We are interested in not only offering assistance based on what is *probable*, but assistance that is beneficial for the user. Utility theory is concerned with the problem of making informed decisions, taking into account all preferences and factors affecting the decisions, and assessing the *utility* of all the outcomes of our decisions. Utility theory has been used in such diverse domains as graphics rendering [22], display of information for time-critical decision making [21], prioritization of repairs [8], and categorization [23].

3.1 Approach

Brown, et al. state that to ascribe user intent, we must identify the salient characteristics of our domain environment and specifically determine goals a user is trying to achieve and the actions to achieve those goals [10]. Social scientists use intentions (as determined by surveying subjects) to measure possible future behaviors (i.e., actions) [31]. That is, what a user says they intend to do is indicative of what they really might do. They note intentions do not necessarily translate into action. Brown, et al., on the other hand, observe behavior (and other environmental events), in an attempt to predict a user's intent so as to predict future behavior.

This approach is based on the belief that what a user intends to do in an environment is the result of environmental stimuli (i.e., events) occurring in the environment, and by the goals they are trying to obtain as a reaction to stimuli. These goals can be explicit (e.g., landing an aircraft) or implicit (e.g., reduce work load). To achieve a goal, a user must perform certain actions to achieve the goal. Goals can be composed of multiple actions, with many pre- and post-conditions. Pre-conditions include directly observable events in the environment (e.g., the plane is going to crash) as well as indirectly observable events (e.g., an increase in the user's cognitive load). These pre-conditions cause a user to pursue a goal. We can use a directed acyclic graph to show causality between the stimuli, goals, and actions. For example, Figure 1 shows three goals — auto pilot landing, manual pilot landing, and reduce cognitive load. Pre-conditions in this figure are the roots of the tree and the actions (e.g., activate ILS) are the leaves of the tree.

There are several advantages to representing users' intentions in a directed acyclic graph, such as the following:

- Goal abstraction allows us to design and detect higher level goals, in pursuit of lower level goals.
- Evidence can be easily and intuitively added and removed (in the form of pre- and post-conditions) as a user interacts with system.
- Pre- and post-conditions for goals and actions are explicitly stated.
- Keyhole plan recognition[1] is made easier by explicitly enumerating atomic actions composing goals [1, 39].
- Natural language explanations of actions based on prediction of goals can be easily generated.

Benyon and Murray use the term *task* or *intentional level* to describe the component of a user model containing knowledge about the user's goals [7]. The task level knowledge is used to infer what goals the user is pursuing. Benyon and Murray state "failure to recognise the intentions underlying some user action will result in less satisfactory interaction" as a result of failing to recognize the pursuit of one goal versus another.

3.2 Architecture

Prediction of user intent is inherently uncertain. Knowledge representations that can dynamically capture and model uncertainty in human-computer interaction as well as the causal relationships between goals, environmental stimuli, and actions as described above are needed to effectively model users.

[1] Plan recognition is the task of ascribing intentions about plans to an agent (human or software), based on observation of the agent's actions. With keyhole plan recognition, the agent is unaware of or indifferent to the plan recognition process.

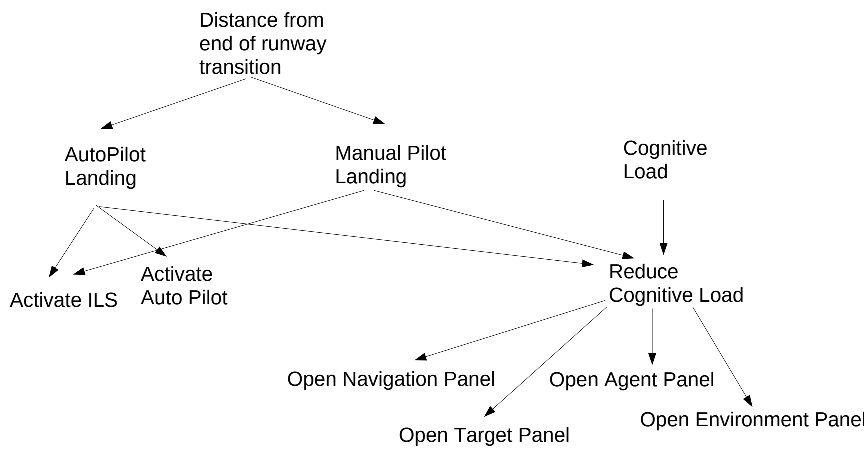

Fig. 1. A directed acyclic graph representation of a user model.

One knowledge representation well suited to representing uncertainty is a Bayesian Network [30]. Bayesian Networks are a probabilistic knowledge representation used to represent uncertain information. The directed acyclic graph structure of the network contains representations of both the conditional dependencies and independencies between elements of the problem domain. The knowledge is represented by nodes called random variables (RVs) and arcs representing the (causal) relationships between variables. The strengths of the relationships are described using parameters encoded in conditional probability tables. A Bayesian network is a mathematically correct and semantically sound model for representing uncertainty, providing a means to show probabilistic relationships between the RVs — in this case goals, actions, and stimuli.

Our Core Interface Agent Architecture (CIaA) is shown in Figure 2. The main component of the architecture is our Bayesian network-based user model. The network consists of three different RV types — goal, action, and pre-condition. Each action RV has an associated utility function used to determine the expected utility of offering assistance to achieve the parent goal. The utility functions and their relationship to ascribing user intent are discussed in the next section. Pre-condition RVs are used to capture environmental events (e.g., psychological factors).

For each observable event in the environment (e.g., button press, altitude change, user is confused, etc.), the host application sends a message, via the message passing interface[2], to the interface agent. This observable is modeled in the user model as either an action or pre-condition. The evaluator stores the observed event in a observation history stack (i.e., most recent event is on top of the history stack).

[2] We use the knowledge query and manipulation language (KQML) [29] due to its general acceptance in the agent community.

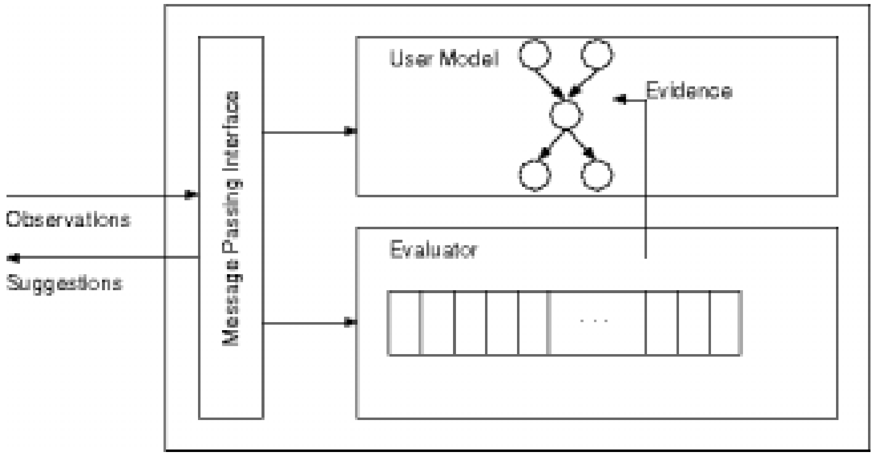

Fig. 2. The Core Interface Agent Architecture.

Periodically (currently once a second) or when new observations arrive from the host application, the interface agent determines if the user needs assistance. The determiniation is based on two factors: the expected utility of offering assistance to acheive a goal and user defined assistance thresholds. The interface agent performs Bayesian network belief updating on all of the goal random variables. All of the relevant observations in the history stack are used as evidence in the Bayesian network. The interface agent then uses the updated probabilities and the utility functions associated at the action RVs to calculate the expected utility of each goal.

The two user defined thresholds, one for offering assistance and one to autonomously perform actions on the user's behalf to obtain a user's goal, determine how/if the interface agent will offer assistance. This approach is the same as Maes [28], except she bases her thresholds on statistical probabilities and we base ours on the expected utility function. The interface agent autonomously performs the actions associated with the highest ranked goal (based on its expected utility) if the goal's expected utility is greater than the autonomy threshold. Else, if the goal's expected utility is above the suggestion threshold, the interface agent sends a request to the user to perform the goal's actions on the user's behalf. Otherwise, the interface agent offers no assistance. Additionally, the interface agent calculates the values of the requirement metrics to determine if the interface agent needs to correct the user model to more accurately reflect the user's intent.

3.3 User Model Design

Our utility theory-based approach addresses the aforementioned deficiencies of statistical-based agents. As mentioned briefly above, we determine the expected utility of offering assistance for a goal, based not only on the probability of an action as determined by performing belief updating of the Bayesian Network, but on the utility of performing the action given the user is pursuing the goal. Let $\mathcal{G}$ be a finite collection of goals. For each goal $G \in \mathcal{G}$ there will be a finite collection of actions A_G for achieving that goal. Let Δ denote the set of applicable discriminators that may also impact the utility (e.g., cognitive load, skill, user preferences). Let $\mathbf{E}$ denote any observed evidence and ξ any background information on the user. Given an action $a \in A_G$, let $U(G, a, \Delta)$ be a positive real-valued utility function, denoting the utility of action a with respect to goal G. Let $Pr(a|\mathbf{E}, \xi)$ denote the probability of action a given evidence E and ξ. Let our expected utility function, $EU(G, \Delta)$, be defined as

$$EU(G, \Delta) = \sum_{a \in A_G} \Pr(a|\mathbf{E}, \xi) U(G, a, \Delta). \tag{2}$$

The interface agent suggests the goal with the greatest expected utility taking into account the user chosen assistance thresholds. Utility theory, using Bayesian techniques for assessing the probabilities, is a non-ad hoc approach for predicting user intent. The utility function $U(G, a, \Delta)$ can take into account *relevancy* of the goal with respect to any number of metrics and/or discriminators in the environment. These metrics tell us what is important, explicitly enumerating those factors that impact the utility of choosing the goal. Δ may take into account the psychological factors, such as the user's cognitive load or preferences; system factors, such as processor load, explicit requirements placed on the system by the designer (e.g., reaction time); or simply the goal at hand[3].

We can use our utility function to determine what actions to take — the ones with the highest expected utility, when to take them — when the expected utility is above some chosen threshold, why to take an action — the action helps the user achieve the goal they are pursuing, and how to take it — the action itself. Key to this determination is realizing metrics that capture relevancy to an action, given the current state of the world.

Constructing the Bayesian network user model and associated utility functions is a difficult research question. For environments where the user's goals and actions are relatively static, we can use any number of well known knowledge elicitation techniques [13]. An approach for determining the goals, actions, and pre- and post-conditions within an environment for adaptive systems is one offered by Benyon and Murray [7]. They point out five analysis phases that must be considered when designing adaptive systems. The first two, functional and data analysis, are analogous to the software engineering techniques of function-oriented and state-oriented problem analysis [14]. Task knowledge analysis focuses on cognitive characteristics required of users by the system (e.g., cognitive loading). User analysis determines the scope of the user population where the system is able to respond. This analysis is concerned with obtaining attributes of users of the application. Environmental analysis is, obviously, concerned with

[3] The current implementation uses a user profile consisting of psychological factors that include spatial and temporal memory, domain expertise, and perceptual acuity.

the environment within which the system is to be situated in, including physical aspects of the system.

However, certain environments do not allow us to fully specify the user model a priori. In these cases, we must realize that our user model must be dynamic. We therefore look to research on dynamic Bayesian networks.

Some researchers take an expert systems approach, where the utility functions are elicited from knowledge experts [21]; some preliminary results for determining utility functions from users exist [15, 17, 35]. However, typically, and certainly in our case, the utility functions are very much determined by individual users. We therefore desire to capture the user-specific utility functions. However, because our user model is dynamic and therefore changes over time (with the possibility of adding and removing goals, actions, pre-conditions), we must be able to specify utility functions dynamically also. It should be obvious we would prefer not to query the user for his/her utility of a particular action.

Therefore, we must determine heuristics for determining the utility functions dynamically. Currently, we feel the user performs the most utilitarian actions first to achieve a goal. Using this simplistic approach, for a given action $a \in A_G$, we can proportionally rank $U(G, a, \Delta)$ based on the order in which the user performs the various actions, normalizing the utility functions.

4 User Models in Practice

Many researchers have used restricted domains (e.g., interface agents for e-mail and news readers) [27] as application domains for their interface agents. While the interface agents used in these domains may be adequate, scalability of the methodologies and techniques used to more complex domains is a problem. We are concerned with integrating intelligent interface agents into complex and dynamic environments, thereby possibly revealing insights into interface agent research not previously recognized with restricted domains.

Our own research in the field of intelligent interface agents is demonstrated by our integration into an expert system shell called PESKI [18, 19, 9], a virtual spaceplane environment [37], and a natural language interface database query system.

PESKI (Probabilities, Expert Systems, Knowledge, and Inference) is an integrated probabilistic knowledge-based expert system shell. PESKI provides users with knowledge acquisition [32], verification and validation [33, 6], data mining [36], and inference engine tools [34], each capable of operating in various communication modes. For more information on PESKI, see the United States Air Force Institute of Technology's Artificial Intelligence Laboratory web site[4]. PESKI was used for our initial tests concerning implementation and usability of our intelligent interface agent [3].

The Virtual SpacePlane (VSP) is a prototype of the Manned SpacePlane (MSP), a spacecraft capable of supporting the United States Air Force's mission

[4] `http://www.afit.af.mil/Schools/EN/ENG/LABS/AI/`

of providing worldwide deployment of space assets, with minimal preflight and in-orbit support from a mission control center. The goals of the VSP project are to uncover, develop and validate the MSP's user interface requirements, develop a prototype virtual spaceplane to demonstrate MSP missions, and to conduct preliminary training experiments. The VSP environment is an accurate, high fidelity presentation of the ground, the Earth's surface as seen from orbit, and the contents of the space environment. The architectural design of the VSP allows for rapid prototyping of the cockpit's user interface and flight dynamics.

Our interface agent is currently being integrated into the VSP to support VSP assistance such as real-time information visualization of real-time data and automation of the landing sequence. Figure 3 shows the preliminary integration of the interface agent within the VSP environment. Here, the interface agent has suggested the user land at Edwards Air Force Base. The suggestion is based on a number of observable environmental stimuli. If the user chooses to allow the agent to achieve this goal (by clicking on the "ok" button in the agent panel), the agent performs the necessary actions to land the spaceplane.

The Clavin System[5] project is a natural language interface database query system. Our CIaA is responsible for adding context to the spoken queries. For example, if the user asks "Which missile hit the plane" the system returns information about F-16s and F-22s and Stinger missiles; then the user's next query asks "What is the cost of these planes", the interface agent determines the user is requesting information specifically about F-16s and F-22s. The use of the interface agent in this system is different from the other two systems in that the user does not perform explicit actions per se in the environment that are observable. However, the robustness of our architecture and knowledge representation allow us to model the spoken queries from the user as well as the return answers to the database queries to effectively help the user.

5 Related Work

Several authors have investigated the use of probabilistic approaches, including Bayesian networks, for plan recognition and generation.

Kirman, et al. investigate the use of a Bayesian network of the world — in the authors' case, a simple room with a mobile robot and a target to be found — and a utility measure on world states to generate plans (sequences of actions) with high expected utility [26]. The number of plans investigated are restricted to those with high utility with respects to attaining a goal. In the authors' work, the goals are known with certainty and the plans to achieve the goals must be found. They show that by using decision-theoretic methods, they can drastically reduce the number of plans that must be investigated.

Waern uses keyhole plan recognition to determine what a user is doing within a route guidance domain and Internet news reader [39]. She uses pre-compiled

[5] The system is named after Cliff Clavin from the T.V. show *Cheers*. Cliff Clavin was known for being a know-it-all.

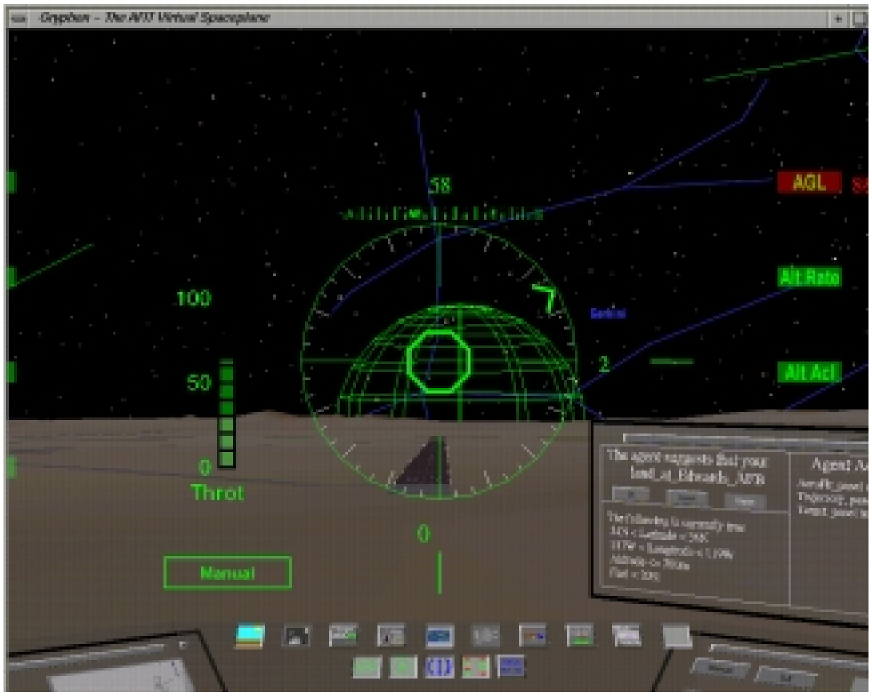

Fig. 3. The Virtual SpacePlane with an Interface Agent.

plans, called *recipes*. Assistance is offered for attaining a goal based on the probability the user is pursuing the goal. Her approach does *not* address the utility of offering assistance. She only considers the probability that offering to perform an action will help the user obtain a goal. Furthermore, her approach is not dynamic; that is, there is no adaptation of the pre-compiled plans. Lastly, there is no way to *trace* the user's execution of a plan nor offer explanations of assistance.

Conati, et al. present a system, ANDES, for long-term knowledge-assessment, plan recognition, and prediction of students' action during physics problem solving [12]. ANDES uses Bayesian networks constructed dynamically from static knowledge bases of specific physics problems as well as physics concepts, laws, formulas, etc. The plan recognition component is used to tailor support for the students when they reach impasses in their problem solving process. Their approach uses explicitly stated goal and problem solving strategy nodes to model the user's problem solving, but does *not* use a utility-based approach. The authors note they are researching methods to expand the fixed knowledge based used to construct the Bayesian networks; in particular, they are adding knowledge to determine common "misconceptions" within the domain and changing the static parameters.

The Pilot's Associate Program was a research effort using plan recognition within a real-time domain to determine goals from actions [4]. The Pilot's Associate was a software agent providing assistance — in the form of wanted and needed information at the correct time — to a combat pilot. They used AND/OR plan-and-goal trees with an associated "dictionary form" for explanations of plans a user was pursuing, and offered assistance to help the pilot. The graph's hierarchical structure produces a common planning language for use among heterogeneous modules. One of the main problems with their approach was due to the technology available at the time for representing and reasoning about uncertainty. Their approach did not account for the uncertainty nor utility in obtaining goals by performing actions. As stated in the authors' conclusions, Pilot's Associate pushed the envelope in uncertainty knowledge representation and reasoning techniques.

The intelligent multimedia presentation systems (IMMPS) project [25] presents an approach to determine, within its domain, the what, when, why, and how, to adapt the system's presentation. Central to their approach is an explicit decomposition of the adaptation process into adaptivity constituents (the "what"), determinants (the "when"), goals (the "why"), and rules (the "how"). They use a decision-theoretic approach to determine what adaptation is best for the current user context (e.g., the user is confused). Their approach does not address agent-based environments since the project is mainly concerned with information presentation and not with processing information from the user. However, their approach appears to be extensible to an agent-based paradigm. There is no way of determining whether a particular method of adaptation, the "how", is feasible within their approach, nor its impact. We believe this places an unnecessary burden on the application designer to account for this and/or limits the allowable adaptations.

6 Issues, Future Research, and Conclusions

In this paper, we described our utility theory-based user model for interface agents. Our approach explicitly models user intent by identifying the goals a user is trying to achieve, with the associated actions to achieve those goals, as well as the pre-conditions that cause a user to pursue the goals. The use of utility theory allows our interface agent to not only reason about the statistical probability that a user is pursuing a goal, but the utility of offering assistance to achieve that goal. Our utility function can use any relevant factor to calculate the utility of offering assistance for a user's goal and we can use our utility function to determine what actions to take, when to take them, why to take an action, and how to take it. The underlying knowledge representation, a Bayesian Network, is capable of representing uncertainty and is dynamic. That is, we can correct the user model over time.

Future efforts propose to provide tools to developers for constructing interface agent user models. Current agent development environments focus on the collaboration and autonomy requirements of an agent (more the former than the latter), while ignoring the adaptivity and robustness requirements. We propose to address these issues explicitly within our development environment, while additionally concentrating on environment specification and agent knowledge base and reasoning mechanisms.

References

1. David W. Albrecht, Ingrid Zukerman, Ann E. Nicholson, and Ariel Bud. Towards a bayesian model for keyhole plan recognition in large domains. In Anthony Jameson, Cécil Paris, and Carlo Tasso, editors, *Proceedings of the Sixth International Conference on User Modeling (UM '97)*, pages 365–376. SpringerWien New York, 1997.
2. Ronald M. Baecker, Jonathan Grudin, William A. S. Buxton, and Saul Greenberg. From customizable systems to intelligent agents. In *Readings in Human-Computer Interaction: Toward the Year 2000*, chapter 12, pages 783–792. Morgan Kaufmann, second edition, 1995.
3. Sheila B. Banks, Robert A. Harrington, Eugene Santos Jr., and Scott M. Brown. Usability testing of an intelligent interface agent. In *Proceedings of the Sixth International Interfaces Conference (Interfaces 97)*, pages 121–123, May 1997.
4. Sheila B. Banks and Carl S. Lizza. Pilot's associate: A cooperative, knowledge-based system application. *IEEE Expert*, pages 18–29, June 1991.
5. Sheila B. Banks, Martin R. Stytz, Eugene Santos Jr., and Scott M. Brown. User modeling for military training: Intelligent interface agents. In *Proceedings of the 19th Interservice/Industry Training Systems and Education Conference*, pages 645–653, December 1997.
6. David Bawcom. An incompleteness handling methodology for validation of bayesian knowledge bases. Master's thesis, Air Force Institute of Technology, 1997.
7. D. Benyon and D. Murray. Adaptive systems: from intelligent tutoring to autonomous agents. *Knowledge-Based Systems*, 6(4):197–219, December 1993.
8. Jack Breese and David Heckerman. Decision-theoretic troubleshooting: A framework for repair and experiment. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 124–132, 1996.
9. Scott M. Brown, Eugene Santos Jr., and Sheila B. Banks. A dynamic bayesian intelligent interface agent. In *Proceedings of the Sixth International Interfaces Conference (Interfaces 97)*, pages 118–120, May 1997.
10. Scott M. Brown, Eugene Santos Jr., Sheila B. Banks, and Mark E. Oxley. Using explicit requirements and metrics for interface agent user model correction. In *Proceedings of the Second International Conference on Autonomous Agents (Agents '98)*, May 1998. to appear.
11. D. N. Chin. Intelligent interfaces as agents. In J. W. Sullivan and S. W. Tyler, editors, *Intelligent User Interfaces.* ACM, New York, 1991.
12. Cristina Conati, Abigail S. Gertner, Kurt VanLehn, and Marek J. Druzdzel. On-line student modeling for coached problem solving using Bayesian networks. In Anthony Jameson, Cécile Paris, and Carlo Tasso, editors, *User Modeling: Proceedings of the Sixth International Conference, UM97*, pages 231–242. Springer Wien New York, Vienna, New York, 1997. Available from http://um.org.
13. Nancy J. Cooke. Varieties of knowledge elicitation techniques. *International Journal of Human-Computer Studies*, 41(6):801–849, 1994.

14. Alan M. Davis. *Software Requirements: Objects, Functions & States.* P T R Prentice Hall, 1993.
15. Marek J. Druzdzel and L. van der Gaag. Elicitation of probabilities for belief networks: Combining qualitative and quantitative information. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 141–148, 1995.
16. Leonard Newton Foner. Paying attention to what's important: Using focus of attention to imporve unsupervised learning. Master's thesis, Massachusetts Institute of Technology, June 1994.
17. Vu Ha and Peter Haddawy. Problem-focused incremental elicitation of multi-attribute utility models. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 215–222, 1997.
18. Robert A. Harrington, Sheila Banks, and Eugene Santos Jr. Development of an intelligent user interface for a generic expert system. In Michael Gasser, editor, *Online Proceedings of the Seventh Midwest Artificial Intelligence and Cognitive Science Conference*, 1996. Available at http://www.cs.indiana.edu/event/maics96/.
19. Robert A. Harrington, Sheila Banks, and Eugene Santos Jr. GESIA: Uncertainty-based reasoning for a generic expert system intelligent user interface. In *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, pages 52–55, 1996.
20. Eric Horvitz. Agents with beliefs: Reflections on Bayesian methods for user modeling. In Anthony Jameson, Cécile Paris, and Carlo Tasso, editors, *User Modeling: Proceedings of the Sixth International Conference, UM97*, pages 441–442. Springer Wien New York, Vienna, New York, 1997. Available from http://um.org.
21. Eric Horvitz and Matthew Barry. Display of information for time-critical decision making. In *Proceedings of the Eleventh Uncertainty in Artificial Intelligence*, pages 296–305, 1995.
22. Eric Horvitz and J. Lengyel. Perception, attention, and resources: A decision-theoretic approach to graphics rendering. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, August 1997.
23. Eric J. Horvitz and Adrian C. Klein. Utility-based abstraction and categorization. In David Heckerman and Abe Mamdani, editors, *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1993.
24. Anthony Jameson. Numeric uncertainty management in user and student modeling: An overview of systems and issues. *User Modeling and User-Adapted Interactions*, 5:193–251, 1995.
25. Charalampos Karagiannidis, Adamatios Koumpis, and Constantine Stephanidis. Deciding 'what', 'when', 'why', and 'how' to adapt in intelligent multimedia presentation systems. In G.P. Faconti and T. Rist, editors, *Proceedings of the Twelvth European Conference on Artificial Intelligence Workshop "Towards a Standard Reference Model for Intelligent Multimedia Presentation Systems"*. John Wiley & Sons, Ltd., August 1996.
26. Jak Kirman, Ann Nicholson, Moises Lejter, Thomas Dean, and Eugene Santos Jr. Using goals to find plans with high expected utility. In *Proceedings of the Second European Workshop on Planning*, pages 158–170, Linkoping, Sweden, 1993.
27. Patti Maes. Modeling adaptive autonomous agents. *Artificial Life Journal*, 1(1 & 2), 1994. MIT Press (C. Langton, Ed.).
28. Pattie Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):811–821, July 1994.

29. James Mayfield, Yannis Labrou, and Tim Finin. Evaluation of kqml as an agent communication language. In Michael J. Woolridge, Jörg P. Müller, and Milind Tambe, editors, *Intelligent Agents II: Agent Theories, Architectures, and Languages*, pages 347–360. Berlin: Springer, 1996.
30. Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.
31. H. Frances G. Pestello and Fred P. Pestello. Ignored, neglected, and abused: The behavior variable in attitude-behavior research. *Symbolic Interaction*, 14(3):341–351, 1991.
32. Eugene Santos Jr., Darwyn O. Banks, and Sheila B. Banks. MACK: A tool for acquiring consistent knowledge under uncertainty. In *Proceedings of the AAAI Workshop on Verification and Validation of Knowledge-Based Systems*, pages 23–32, 1997.
33. Eugene Santos Jr., Howard T. Gleason, and Sheila B. Banks. BVAL: Probabilistic knowledge-base validation. In *Proceedings of the AAAI Workshop on Verification and Validation of Knowledge-Based Systems*, pages 13–22, 1997.
34. Solomon Eyal Shimony, Carmel Domshlak, and Eugene Santos Jr. Cost-sharing heuristic for bayesian knowledge-bases. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 421–428, 1997.
35. Yoav Shoham. Conditional utility, utiltiy independence, and utility networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 429–436, 1997.
36. Daniel J. Stein III, Sheila B. Banks, Eugene Santos Jr., and Michael L. Talbert. Utilizing goal-directed data mining for incompleteness repair in knowledge bases. In Eugene Santos Jr., editor, *Proceedings of the Eighth Midwest Artificial Intelligence and Cognitive Science Conference*, pages 82–85. AAAI Press, 1997.
37. Martin R. Stytz and Sheila B. Banks. The virtual spaceplane: A modeling and simulation tool for advanced prototyping, requirements development, and training for the manned spaceplane project. In *Proceedings of the 19th Interservice/Industry Training Systems and Education Conference*, December 1997.
38. Katia Sycara, Keith Decker, Anandeep Pannu, Mike Williamson, and Dajun Zeng. Distributed intelligent agents. *IEEE Expert*, 11(6):36–46, December 1996.
39. Annika Waern. *Recognising Human Plans: Issues for Plan Recognition in Human-Computer Interaction*. PhD thesis, Royal Institute of Technology, 1996.